XP002937254 C29

10 CHANNEL CONTROL

PD: 11 - 1999
P: 95 - 126 = (32)

10.1 SCOPE

This section describes how the channel of a piconet is established and how units can be added to and released from the piconet. Several states of operation of the Bluetooth units are defined to support these functions. In addition, the operation of several piconets sharing the same area, the so-called scatternet, is discussed. A special section is attributed to the Bluetooth clock which plays a major role in the FH synchronization.

10.2 MASTER-SLAVE DEFINITION

The channel in the piconet is characterized entirely by the master of the piconet. The Bluetooth device address (BD_ADDR) of the master determines the FH hopping sequence and the channel access code; the system clock of the master determines the phase in the hopping sequence and sets the timing. In addition, the master controls the traffic on the channel by a polling scheme.

By definition, the **master** is represented by the Bluetooth unit that initiates the connection (to one or more **slave** units). Note that the names 'master' and 'slave' only refer to the protocol on the channel: the Bluetooth units themselves are identical; that is, any unit can become a master of a piconet. Once a piconet has been established, master-slave roles can be exchanged. This is described in more detail in Section 10.9.3 on page 123.

10.3 BLUETOOTH CLOCK

Every Bluetooth unit has an internal system clock which determines the timing and hopping of the transceiver. The Bluetooth clock is derived from a free running native clock which is never adjusted and is never turned off. For synchronization with other units, only offsets are used that, added to the native clock, provide temporary Bluetooth clocks which are mutually synchronized. It should be noted that the Bluetooth clock has no relation to the time of day; it can therefore be initialized at any value. The Bluetooth clock provides the heart beat of the Bluetooth transceiver. Its resolution is at least half the TX or RX slot length, or 312.5 µs. The clock has a cycle of about a day. If the clock is implemented with a counter, a 28-bit counter is required that wraps around at 2^{28}-1. The LSB ticks in units of 312.5 µs, giving a clock rate of 3.2 kHz.

The timing and the frequency hopping on the channel of a piconet is determined by the Bluetooth clock of the master. When the piconet is established, the master clock is communicated to the slaves. Each slave adds an offset to its native clock to be synchronized to the master clock. Since the clocks are free-running, the offsets have to be updated regularly.

The clock determines critical periods and triggers the events in the Bluetooth receiver. Four periods are important in the Bluetooth system: 312.5 µs, 625 µs, 1.25 ms, and 1.28 s; these periods correspond to the timer bits CLK_0, CLK_1, CLK_2, and CLK_{12}, respectively, see Figure 10.1 on page 96. Master-to-slave transmission starts at the even-numbered slots when CLK_0 and CLK_1 are both zero.

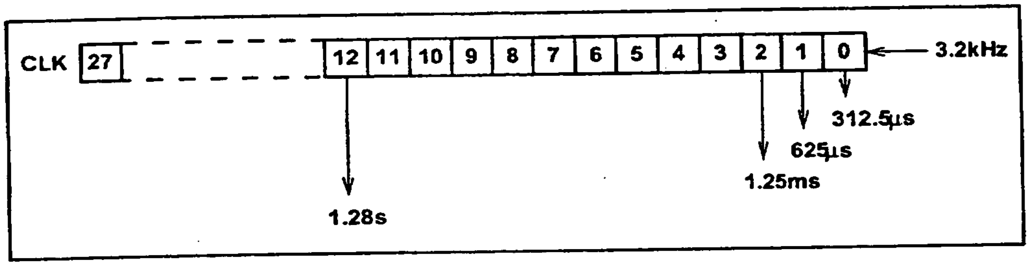

Figure 10.1: Bluetooth clock.

In the different modes and states a Bluetooth unit can reside in, the clock has different appearances:

- CLKN native clock
- CLKE estimated clock
- CLK master clock

CLKN is the free-running native clock and is the reference to all other clock appearances. In states with high activity, the native clock is driven by the reference crystal oscillator with worst case accuracy of +/-20ppm. In the low power states, like **STANDBY, HOLD, PARK**, the native clock may be driven by a low power oscillator (LPO) with relaxed accuracy (+/-250ppm).

CLKE and CLK are derived from the reference CLKN by adding an offset. CLKE is a clock estimate a paging unit makes of the native clock of the recipient; i.e. an offset is added to the CLKN of the pager to approximate the CLKN of the recipient, see Figure 10.2 on page 97. By using the CLKN of the recipient, the pager speeds up the connection establishment.

CLK is the master clock of the piconet. It is used for all timing and scheduling activities in the piconet. All Bluetooth devices use the CLK to schedule their transmission and reception. The CLK is derived from the native clock CLKN by adding an offset, see Figure 10.3 on page 97. The offset is zero for the master since CLK is identical to its own native clock CLKN. Each slave adds an appropriate offset to its CLKN such that the CLK corresponds to the CLKN of the master. Although all CLKNs in the Bluetooth devices run at the same nominal rate, mutual drift causes inaccuracies in CLK. Therefore, the offsets in the slaves must be regularly updated such that CLK is approximately CLKN of the master.

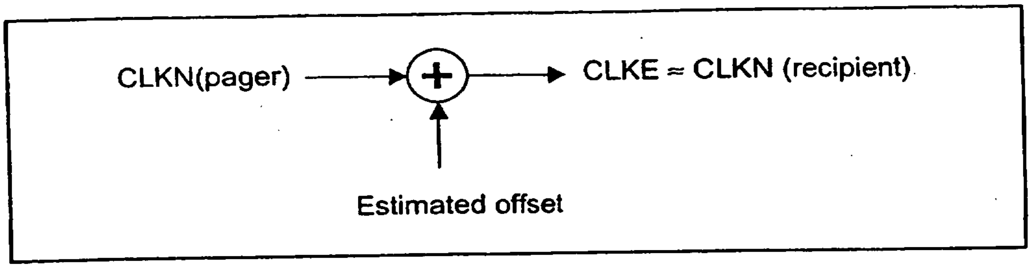

Figure 10.2: Derivation of CLKE

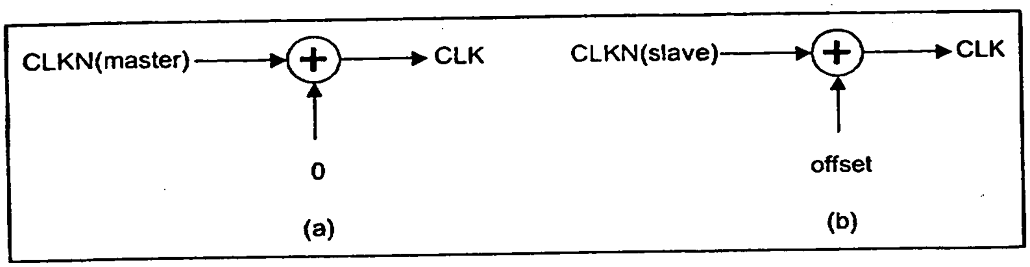

Figure 10.3: Derivation of CLK in master (a) and in slave (b).

10.4 OVERVIEW OF STATES

Figure 10.4 on page 98 shows a state diagram illustrating the different states used in the Bluetooth link controller. There are two major states: **STANDBY** and **CONNECTION**; in addition, there are seven substates, **page**, **page scan**, **inquiry**, **inquiry scan**, **master response**, **slave response**, and **inquiry response**. The substates are interim states that are used to add new slaves to a piconet. To move from one state to the other, either commands from the Bluetooth link manager are used, or internal signals in the link controller are used (such as the trigger signal from the correlator and the timeout signals).

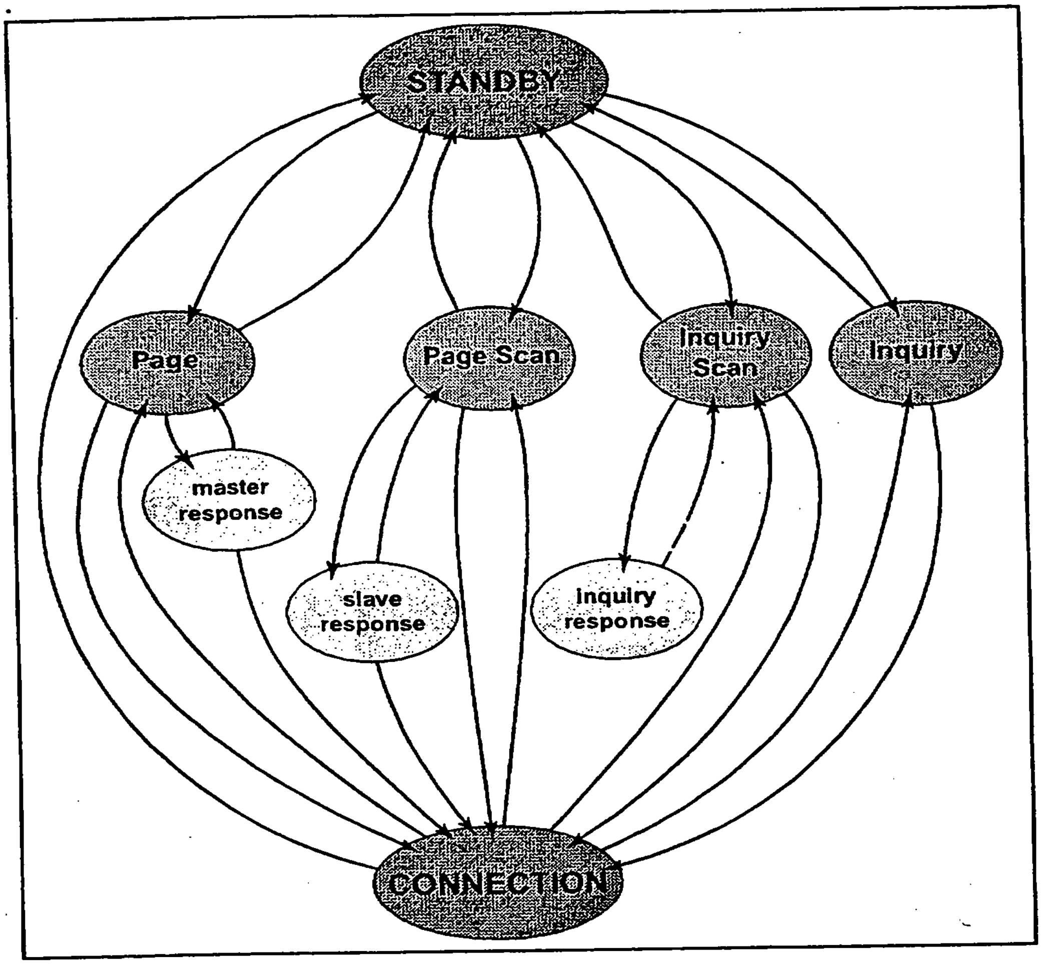

Figure 10.4: State diagram of Bluetooth link controller.

10.5 STANDBY STATE

The **STANDBY** state is the default state in the Bluetooth unit. In this state, the Bluetooth unit is in a low-power mode. Only the native clock is running at the accuracy of the LPO (or better).

The controller may leave the **STANDBY** state to scan for page or inquiry messages, or to page or inquiry itself. When responding to a page message, the unit will not return to the **STANDBY** state but enter the **CONNECTION** state as a slave. When carrying out a successful page attempt, the unit will enter the **CONNECTION** state as a master. The intervals with which scan activities can be carried out are discussed in Section 10.6.2 on page 99 and Section 10.7.2 on page 109.

10.6 ACCESS PROCEDURES

10.6.1 General

In order to establish new connections the procedures inquiry and paging are used. The inquiry procedure enables a unit to discover which units are in range, and what their device addresses and clocks are. With the paging procedure, an actual connection can be established. Only the Bluetooth device address is required to set up a connection. Knowledge about the clock will accelerate the setup procedure. A unit that establishes a connection will carry out a page procedure and will automatically be the master of the connection.

In the paging and inquiry procedures, the device access code (DAC) and the inquiry access code (IAC) are used, respectively. A unit in the **page scan** or **inquiry scan** substate correlates against these respective access codes with a matching correlator.

For the paging process, several paging schemes can be applied. There is one mandatory paging scheme which has to be supported by each Bluetooth device. This mandatory scheme is used when units meet for the first time, and in case the paging process directly follows the inquiry process. Two units, once connected using a mandatory paging/scanning scheme, may agree on an optional paging/scanning scheme. Optional paging schemes are discussed in "Appendix VII" on page 999. In the current chapter, only the mandatory paging scheme is considered.

10.6.2 Page scan

In the **page scan** substate, a unit listens for its own device access code for the duration of the scan window $T_{w\ page\ scan}$. During the scan window, the unit listens at a single hop frequency, its correlator matched to its device access code. The scan window shall be long enough to completely scan 16 page frequencies.

When a unit enters the **page scan** substate, it selects the scan frequency according to the page hopping sequence corresponding to this unit, see Section 11.3.1 on page 135. This is a 32-hop sequence (or a 16-hop sequence in case of a reduced-hop system) in which each hop frequency is unique. The page hopping sequence is determined by the unit's Bluetooth device address (BD_ADDR). The phase in the sequence is determined by $CLKN_{16-12}$ of the unit's native clock ($CLKN_{15-12}$ in case of a reduced-hop system); that is, every 1.28s a different frequency is selected.

If the correlator exceeds the trigger threshold during the **page scan**, the unit will enter the **slave response** substate, which is described in Section 10.6.4.1 on page 105.

The **page scan** substate can be entered from the **STANDBY** state or the **CONNECTION** state. In the **STANDBY** state, no connection has been established and the unit can use all the capacity to carry out the **page scan.** Before entering the **page scan** substate from the **CONNECTION** state, the unit preferably reserves as much capacity for scanning. If desired, the unit may place ACL connections in the HOLD mode or even use the PARK mode, see Section 10.8.3 on page 114 and Section 10.8.4 on page 115. SCO connections are preferably not interrupted by the **page scan.** In this case, the **page scan** may be interrupted by the reserved SCO slots which have higher priority than the **page scan.** SCO packets should be used requiring the least amount of capacity (**HV3** packets). The scan window shall be increased to minimize the setup delay. If one SCO link is present using **HV3** packets and T_{SCO}=6 slots, a total scan window $T_{w\ page\ scan}$ of at least 36 slots (22.5ms) is recommended; if two SCO links are present using **HV3** packets and T_{SCO}=6 slots, a total scan window of at least 54 slots (33.75ms) is recommended.

The scan interval $T_{page\ scan}$ is defined as the interval between the beginnings of two consecutive page scans. A distinction is made between the case where the scan interval is equal to the scan window $T_{w\ page\ scan}$ (continuous scan), the scan interval is maximal 1.28s, or the scan interval is maximal 2.56s. These three cases determine the behavior of the paging unit; that is, whether the paging unit shall use R0, R1 or R2, see also Section 10.6.3 on page 101. Table 10.1 illustrates the relationship between $T_{page\ scan}$ and modes R0, R1 and R2. Although scanning in the R0 mode is continuous, the scanning may be interrupted by for example reserved SCO slots. The scan interval information is included in the SR field in the FHS packet.

During page scan the Bluetooth unit may choose to use an optional scanning scheme. (An exception is the page scan after returning an inquiry response message. See Section 10.7.4 on page 111 for details.)

SR mode	$T_{page\ scan}$	N_{page}
R0	continuous	≥1
R1	≤ 1.28s	≥ 128
R2	≤ 2.56s	≥ 256
Reserved		

Table 10.1: Relationship between scan interval, train repetition, and paging modes R0, R1 and R2.

10.6.3 Page

The **page** substate is used by the master (source) to activate and connect to a slave (destination) which periodically wakes up in the **page scan** substate. The master tries to capture the slave by repeatedly transmitting the slave's device access code (DAC) in different hop channels. Since the Bluetooth clocks of the master and the slave are not synchronized, the master does not know exactly when the slave wakes up and on which hop frequency. Therefore, it transmits a train of identical DACs at different hop frequencies, and listens in between the transmit intervals until it receives a response from the slave.

The page procedure in the master consists of a number of steps. First, the slave's device address is used to determine the page hopping sequence, see Section 11.3.2 on page 135. This is the sequence the master will use to reach the slave. For the phase in the sequence, the master uses an estimate of the slave's clock. This estimate can for example be derived from timing information that was exchanged during the last encounter with this particular device (which could have acted as a master at that time), or from an inquiry procedure. With this estimate CLKE of the slave's Bluetooth clock, the master can predict when the slave wakes up and on which hop channel.

The estimate of the Bluetooth clock in the slave can be completely wrong. Although the master and the slave use the same hopping sequence, they use different phases in the sequence and will never meet each other. To compensate for the clock drifts, the master will send its page message during a short time interval on a number of wake-up frequencies. It will in fact transmit also on hop frequencies just before and after the current, predicted hop frequency. During each TX slot, the master sequentially transmits on two different hop frequencies. Since the page message is the ID packet which is only 68 bits in length, there is ample of time (224.5 μs minimal) to switch the synthesizer. In the following RX slot, the receiver will listen sequentially to two corresponding RX hops for ID packet. The RX hops are selected according to the page_response hopping sequence. The page_response hopping sequence is strictly related to the page hopping sequence; that is: for each page hop there is a corresponding page_response hop. The RX/TX timing in the page sub-

state has been described in Section 9, see also Figure 9.4 on page 91. In the next TX slot, it will transmit on two hop frequencies different from the former ones. The synthesizer hop rate is increased to 3200 hops/s.

A distinction must be made between the 79-hop systems and the 23-hop systems. First the 79-hop systems are considered. With the increased hopping rate as described above, the transmitter can cover 16 different hop frequencies in 16 slots or 10 ms. The page hopping sequence is divided over two paging trains **A** and **B** of 16 frequencies. Train **A** includes the 16 hop frequencies surrounding the current, predicted hop frequency f(k), where k is determined by the clock estimate $CLKE_{16\text{-}12}$. So the first train consists of hops

f(k-8), f(k-7),...,f(k),...,f(k+7)

When the difference between the Bluetooth clocks of the master and the slave is between -8x1.28 s and +7x1.28 s, one of the frequencies used by the master will be the hop frequency the slave will listen to. However, since the master does not know when the slave will enter the **page scan** substate, he has to repeat this train **A** N_{page} times or until a response is obtained. If the slave scan interval corresponds to R1, the repetition number is at least 128; if the slave scan interval corresponds to R2, the repetition number is at least 256.
Note that $CLKE_{16\text{-}12}$ changes every 1.28 s; therefore, every 1.28 s, the trains will include different frequencies of the page hopping set.

When the difference between the Bluetooth clocks of the master and the slave is less than -8x1.28 s or larger than +7x1.28 s, more distant hops must be probed. Since in total, there are only 32 dedicated wake-up hops, the more distant hops are the remaining hops not being probed yet. The remaining 16 hops are used to form the new 10 ms train **B**. The second train consists of hops

f(k-16), f(k-15),...,f(k-9),f(k+8)...,f(k+15)

Train **B** is repeated for N_{page} times. If still no response is obtained, the first train **A** is tried again N_{page} times. Alternate use of train A and train B is continued until a response is received or the timeout *pageTO* is exceeded. If during one of the listening occasions, a response is returned by the slave, the master unit enters the **master response** substate.

The description for paging and **page scan** procedures given here has been tailored towards the 79-hop systems used in the US and Europe. For the 23-hop systems as used in Japan and some European countries, the procedure is slightly different. In the 23-hop case, the length of the page hopping sequence is reduced to 16. As a consequence, there is only a single train (train **A**) including all the page hopping frequencies. The phase to the page hopping sequence is not $CLKE_{16\text{-}12}$ but $CLKE_{15\text{-}12}$. An estimate of the slave's clock does not have to be made.

The **page** substate can be entered from the **STANDBY** state or the **CONNECTION** state. In the **STANDBY** state, no connection has been established and

the unit can use all the capacity to carry out the page. Before entering the page substate from the CONNECTION state, the unit shall free as much capacity as possible for scanning. To ensure this, it is recommended that the ACL connections are put on hold or park. However, the SCO connections shall not be disturbed by the page. This means that the page will be interrupted by the reserved SCO slots which have higher priority than the page. In order to obtain as much capacity for paging, it is recommended to use the SCO packets which use the least amount of capacity (**HV3** packets). If SCO links are present, the repetition number N_{page} of a single train shall be increased, see Table 10.2. Here it has been assumed that the **HV3** packet are used with an interval T_{SCO}=6 slots, which would correspond to a 64 kb/s voice link.

SR mode	no SCO link	one SCO link (HV3)	two SCO links (HV3)
R0	$N_{page} \geq 1$	$N_{page} \geq 2$	$N_{page} \geq 3$
R1	$N_{page} \geq 128$	$N_{page} \geq 256$	$N_{page} \geq 384$
R2	$N_{page} \geq 256$	$N_{page} \geq 512$	$N_{page} \geq 768$

Table 10.2: Relationship between train repetition, and paging modes R0, R1 and R2 when SCO links are present.

The construction of the page train is independent on the presence of SCO links; that is, SCO packets are sent on the reserved slots but do not affect the hop frequencies used in the unreserved slots, see Figure 10.5 on page 103.

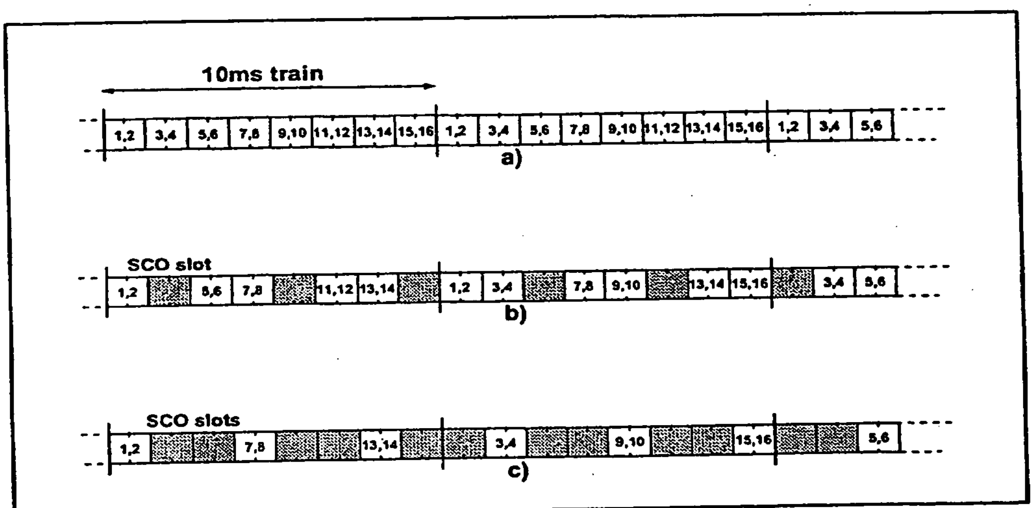

Figure 10.5: Conventional page (a), page while one SCO link present (b), page while two SCO links present (c).

For the descriptions of optional paging schemes see "Appendix VII" on page 999.

10.6.4 Page response procedures

When a page message is successfully received by the slave, there is a coarse FH synchronization between the master and the slave. Both the master and the slave enter a response routine to exchange vital information to continue the connection setup. Important for the piconet connection is that both Bluetooth units use the same channel access code, use the same channel hopping sequence, and that their clocks are synchronized. These parameters are derived from the master unit. The unit that initializes the connection (starts paging) is defined as the master unit (which is thus only valid during the time the piconet exists). The channel access code and channel hopping sequence are derived from the Bluetooth device address (BD_ADDR) of the master. The timing is determined by the master clock. An offset is added to the slave's native clock to temporarily synchronize the slave clock to the master clock. At start-up, the master parameters have to be transmitted from the master to the slave. The messaging between the master and the slave at start-up will be considered in this section.

The initial messaging between master and slave is shown in Table 10.3 on page 104 and in Figure 10.6 on page 105 and Figure 10.7 on page 105. In those two figures frequencies f (k), f(k+1), etc. are the frequencies of the page hopping sequence determined by the slave's BD_ADDR. The frequencies f'(k), f'(k+1), etc. are the corresponding page_response frequencies (slave-to-master). The frequencies g(m) belong to the channel hopping sequence.

Step	Message	Direction	Hopping Sequence	Access Code and Clock
1	slave ID	master to slave	page	slave
2	slave ID	slave to master	page response	slave
3	FHS	master to slave	page	slave
4	slave ID	slave to master	page response	slave
5	1st packet master	master to slave	channel	master
6	1st packet slave	slave to master	channel	master

Table 10.3: Initial messaging during start-up.

In step 1 (see Table 10.3 on page 104), the master unit is in **page** substate and the slave unit in the **page scan** substate. Assume in this step that the page message (= slave's device access code) sent by the master reaches the slave. On recognizing its device access code, the slave enters the **slave response** in step 2. The master waits for a reply from the slave and when this arrives in step 2, it will enter the **master response** in step 3. Note that during the initial message exchange, all parameters are derived from the slave's BD_ADDR, and that only the page hopping and page_response hopping sequences are used

(which are also derived from the slave's BD_ADDR). Note that when the master and slave enter the response states, their clock input to the page and page_response hop selection is frozen as is described in Section 11.3.3 on page 136.

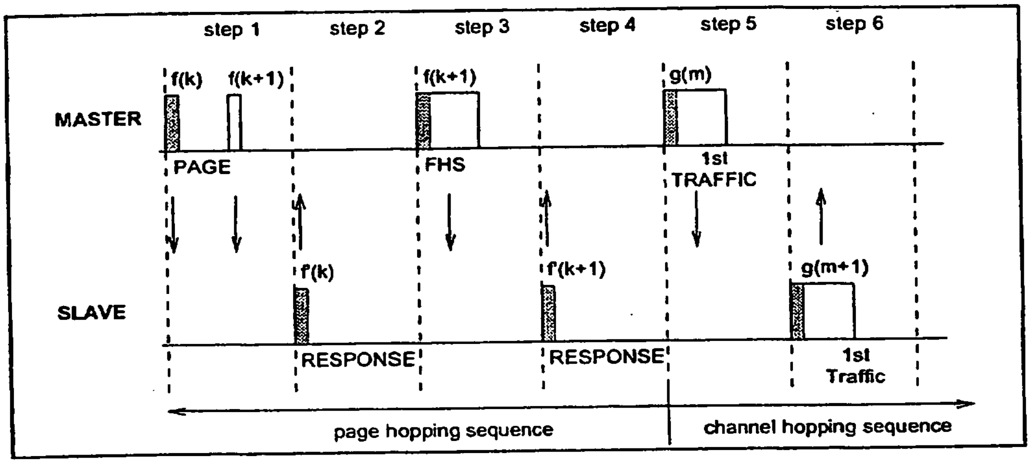

Figure 10.6: Messaging at initial connection when slave responds to first page message.

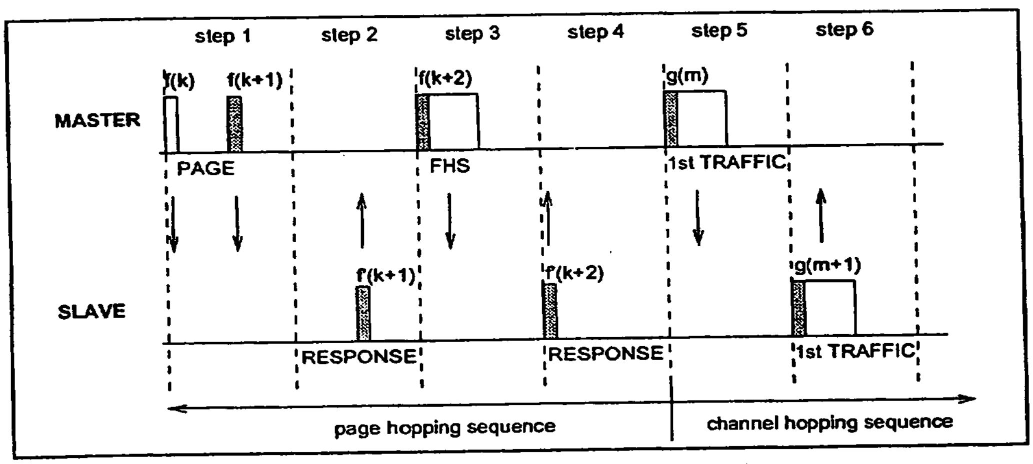

Figure 10.7: Messaging at initial connection when slave responds to second page message.

10.6.4.1 Slave response

After having received its own device access code in step 1, the slave unit transmits a response message in step 2. This response message again only consists of the slave's device access code. The slave will transmit this response 625 µs after the beginning of the received page message (slave ID packet) and at the response hop frequency that corresponds to the hop frequency in which the page message was received. The slave transmission is therefore time

aligned to the master transmission. During initial messaging, the slave still uses the page response hopping sequence to return information to the master. The clock input $CLKN_{16\text{-}12}$ is frozen at the value it had at the time the page message was received.

After having sent the response message, the slave's receiver is activated (312.5 μs after the start of the response message) and awaits the arrival of a **FHS** packet. Note that a **FHS** packet can already arrive 312.5 μs after the arrival of the page message as shown in Figure 10.7 on page 105, and not after 625 μs as is usually the case in the RX/TX timing. More details about the timing can be found in Section 9.6 on page 91.

If the setup fails before the **CONNECTION** state has been reached, the following procedure is carried out. The slave will keep listening as long as no **FHS** packet is received until *pagerespTO* is exceeded. Every 1.25 ms, however, it will select the next master-to-slave hop frequency according to the page hop sequence. If nothing is received after *pagerespTO*, the slave returns back to the **page scan** substate for one scan period. Length of the scan period depends on the SCO slots present. If no page message is received during this additional scan period, the slave will resume scanning at its regular scan interval and return to the state it was in prior to the first page scan state.

If a **FHS** packet is received by the slave in the **slave response** substate, the slave returns a response (slave's device access code only) in step 4 to acknowledge the reception of the **FHS** packet (still using the page response hopping sequence). The transmission of this response packet is based on the reception of the **FHS** packet. Then the slave changes to the channel (master's) access code and clock as received from the **FHS** packet. Only the 26 MSBs of the master clock are transferred: the timing is assumed such that CLK_1 and CLK_0 are both zero at the time the **FHS** packet was received as the master transmits in even slots only. From the master clock in the **FHS** packet, the offset between the master's clock and the slave's clock is determined and reported to the slave's link manager.

Finally, the slave enters the **CONNECTION** state in step 5. From then on, the slave will use the master's clock and the master BD_ADDR to determine the channel hopping sequence and the channel access code. The connection mode starts with a POLL packet transmitted by the master. The slave responds with any type of packet. If the POLL packet is not received by the slave, or the response packet is not received by the master, within *newconnectionTO* number of slots after FHS packet acknowledgement, the master and the slave will return to page and page scan substates, respectively.See Section 10.8 on page 112

10.6.4.2 Master response

When the master has received a response message from the slave in step 2, it will enter the **master response** routine. It freezes the current clock input to the page hop selection scheme. Then the master will transmit a **FHS** packet in step 3 containing the master's real-time Bluetooth clock, the master's 48-bit BD_ADDR address, the BCH parity bits, and the class of device. The **FHS** packet contains all information to construct the channel access code without requiring a mathematical derivation from the master device address. The **FHS** packet is transmitted at the beginning of the master-to-slave slot following the slot in which the slave has responded. So the TX timing of the **FHS** is not based on the reception of the response packet from the slave. The **FHS** packet may therefore be sent 312.5 µs after the reception of the response packet like shown in Figure 10.7 on page 105 and not 625 µs after the received packet as is usual in the RX/TX timing, see also Section 9.6 on page 91.

After the master has sent its **FHS** packet, it waits for a second response from the slave in step 4 which acknowledges the reception of the **FHS** packet. Again this is only the slave's device access code. If no response is received, the master retransmits the **FHS** packet, but with an updated clock and still using the slave's parameters. It will retransmit (the clock is updated every retransmission) until a second slave response is received, or the timeout of *pagerespTO* is exceeded. In the latter case, the master turns back to the **page** substate and sends an error message to the link manager. During the retransmissions of the **FHS** packet, the master keeps using the page hopping sequence.

If the slave's response is indeed received, the master changes to the master parameters, so the channel access code and the master clock. The lower clock bits CLK_0 and CLK_1 are zero at the start of the **FHS** packet transmission and are not included in the **FHS** packet. Finally, the master enters the **CONNECTION** state in step 5. The master BD_ADDR is used to change to a new hopping sequence, the *channel hopping sequence*. The channel hopping sequence uses all 79 hop channels in a (pseudo) random fashion, see also Section 11.3.6 on page 138. The master can now send its first traffic packet in a hop determined with the new (master) parameters. This first packet will be a POLL packet. See Section 10.8 on page 112.

The master can now send its first traffic packet in a hop determined with the new (master) parameters. The first packet in this state is a POLL packet sent by the master. This packet will be sent within *newconnectionTO* number of slots after reception of the FHS packet acknowledgement. The slave will respond with any type of packet. If the POLL packet is not received by the slave or the POLL packet response is not received by the master within *newconnectionTO* number of slots, the master and the slave will return to page and page scan substates, respectively.

10.7 INQUIRY PROCEDURES

10.7.1 General

In the Bluetooth system, an inquiry procedure is defined which is used in applications where the destination's device address is unknown to the source. One can think of public facilities like printers or facsimile machines, or access points to a LAN. Alternatively, the inquiry procedure can be used to discover which other Bluetooth units are within range. During an **inquiry** substate, the discovering unit collects the Bluetooth device addresses and clocks of all units that respond to the inquiry message. It can then, if desired, make a connection to any one of them by means of the previously described page procedure.

The inquiry message broadcasted by the source does not contain any information about the source. However, it may indicate which class of devices should respond. There is one general inquiry access code (GIAC) to inquire for any Bluetooth device, and a number of dedicated inquiry access codes (DIAC) that only inquire for a certain type of devices. The inquiry access codes are derived from reserved Bluetooth device addresses and are further described in Section 4.2.1.

A unit that wants to discover other Bluetooth units enters an **inquiry** substate. In this substate, it continuously transmits the inquiry message (which is the ID packet, see Section 4.4.1.1 on page 55) at different hop frequencies. The **inquiry** hop sequence is always derived from the LAP of the GIAC. Thus, even when DIACs are used, the applied hopping sequence is generated from the GIAC LAP. A unit that allows itself to be discovered, regularly enters the **inquiry scan** substate to respond to inquiry messages. The following sections describe the message exchange and contention resolution during inquiry response. The inquiry response is optional: a unit is not forced to respond to an inquiry message.

10.7.2 Inquiry scan

The **inquiry scan** substate is very similar to the **page scan** substate. However, instead of scanning for the unit's device access code, the receiver scans for the inquiry access code long enough to completely scan for 16 inquiry frequencies. The length of this scan period is denoted $T_{w\_inquiry\_scan}$. The scan is performed at a single hop frequency. As in the page procedure, the inquiry procedure uses 32 dedicated inquiry hop frequencies according to the *inquiry hopping sequence*. These frequencies are determined by the general inquiry address. The phase is determined by the native clock of the unit carrying out the **inquiry scan**; the phase changes every 1.28s.

Instead or in addition to the general inquiry access code, the unit may scan for one or more dedicated inquiry access codes. However, the scanning will follow the inquiry hopping sequence which is determined by the general inquiry address. If an inquiry message is recognized during an inquiry wake-up period, the Bluetooth unit enters the **inquiry response** substate.

The **inquiry scan** substate can be entered from the **STANDBY** state or the **CONNECTION** state. In the **STANDBY** state, no connection has been established and the unit can use all the capacity to carry out the **inquiry scan.** Before entering the **inquiry scan** substate from the **CONNECTION** state, the unit preferably reserves as much capacity as possible for scanning. If desired, the unit may place ACL connections in the HOLD mode or even use the PARK mode, see Section 10.8.3 on page 114. SCO connections are preferably not interrupted by the **inquiry scan**. In this case, the **inquiry scan** may be interrupted by the reserved SCO slots which have higher priority than the **inquiry scan.** SCO packets should be used requiring the least amount of capacity (**HV3** packets). The scan window, $T_{w\ inquiry\ scan}$, shall be increased to increase the probability to respond to an inquiry message. If one SCO link is present using HV3 packets and T_{SCO}=6 slots, a total scan window of at least 36 slots (22.5ms) is recommended; if two SCO links are present using HV3 packets and T_{SCO}=6 slots, a total scan window of at least 54 slots (33.75ms) is recommended.

The scan interval $T_{inquiry\ scan}$ is defined as the interval between two consecutive inquiry scans. The **inquiry scan** interval shall be at most 2.56 s.

10.7.3 Inquiry

The **inquiry** substate is used by the unit that wants to discover new devices. This substate is very similar to the **page** substate, the same TX/RX timing is used as used for paging, see Section 9.6 on page 91 and Figure 9.4 on page 91. The TX and RX frequencies follow the inquiry hopping sequence and the inquiry response hopping sequence, and are determined by the general inquiry access code and the native clock of the discovering device. In between inquiry transmissions, the Bluetooth receiver scans for inquiry response messages. When found, the entire response packet (which is in fact a **FHS** packet) is read, after which the unit continues with the inquiry transmissions. So the Bluetooth unit in an **inquiry** substate does not acknowledge the inquiry response messages. It keeps probing at different hop channels and in between listens for response packets. Like in the **page** substate, two 10 ms trains **A** and **B** are defined, splitting the 32 frequencies of the inquiry hopping sequence into two 16-hop parts. A single train must be repeated for at least $N_{inquiry}$=256 times before a new train is used. In order to collect all responses in an error-free environment, at least three train switches must have taken place. As a result, the **inquiry** substate may have to last for 10.24 s unless the inquirer collects enough responses and determines to abort the inquiry substate earlier. If desired, the inquirer can also prolong the inquiry substate to increase the probability of receiving all responses in an error-prone environment. If an inquiry procedure is automatically initiated periodically (say a 10 s period every minute), then the interval between two inquiry instances must be determined randomly. This is done to avoid two Bluetooth units to synchronize their inquiry procedures.

The **inquiry** substate is continued until stopped by the Bluetooth link manager (when it decides that it has sufficient number of responses), or when a timeout has been reached (*inquiryTO*).

The **inquiry** substate can be entered from the **STANDBY** state or the **CONNECTION** state. In the **STANDBY** state, no connection has been established and the unit can use all the capacity to carry out the inquiry. Before entering the inquiry substate from the **CONNECTION** state, the unit shall free as much capacity as possible for scanning. To ensure this, it is recommended that the ACL connections are put on hold or park. However, the SCO connections shall not be disturbed by the inquiry. This means that the inquiry will be interrupted by the reserved SCO slots which have higher priority than the inquiry. In order to obtain as much capacity for inquiry, it is recommended to use the SCO packets which use the least amount of capacity (**HV3** packets). If SCO links are present, the repetition number $N_{inquiry}$ shall be increased, see Table 10.4 on page 111.
Here it has been assumed that the **HV3** packet are used with an interval T_{SCO}=6 slots, which would correspond to a 64 kb/s voice link.

	no SCO link	one SCO link (HV3)	two SCO links (HV3)
$N_{inquiry}$	≥ 256	≥ 512	≥ 768

Table 10.4: Increase of train repetition when SCO links are present.

10.7.4 Inquiry response

For the inquiry operation, there is only a slave response, no master response. The master listens between inquiry messages for responses, but after reading a response, it continues to transmit inquiry messages. The slave response routine for inquiries differs completely from the slave response routine applied for pages. When the inquiry message is received in the **inquiry scan** substate, a response message containing the recipient's address must be returned. This response message is a conventional **FHS** packet carrying the unit's parameters. However, a contention problem may arise when several Bluetooth units are in close proximity to the inquiring unit and all respond to an inquiry message at the same time. First of all, every Bluetooth unit has a free running clock; therefore, it is highly unlikely that they all use the same phase of the inquiry hopping sequence. However, in order to avoid collisions between units that do wake up in the same inquiry hop channel simultaneously, the following protocol in the slave's **inquiry response** is used. If the slave receives an inquiry message, it generates a random number RAND between 0 and 1023. In addition, it freezes the current input value (phase) to the hop selection scheme, see also Section 11.3.5 on page 137. The slave then returns to the **CONNECTION** or **STANDBY** state for the duration of RAND time slots. Before returning to the **CONNECTION** or **STANDBY** state, the unit may go through the page scan substate; this page scan must use the mandatory page scan scheme. After at least RAND slots, the unit will return to the **inquiry response** substate. On the first inquiry message received the slave returns an **FHS** response packet to the master. If during the scan no trigger occurs within a timeout period of *inqrespTO*, the slave returns to the **STANDBY** or **CONNECTION** state. If the unit does receive an inquiry message and returns an FHS packet, it adds an offset of 1 to the phase in the inquiry hop sequence (the phase has a 1.28 s resolution) and enters the **inquiry scan** substate again. If the slave is triggered again, it repeats the procedure using a new RAND. The offset to the clock accumulates each time a **FHS** packet is returned. During a 1.28 s probing window, a slave on average responses 4 times, but on different frequencies and at different times. Possible SCO slots should have priority over response packets; that is, if a response packet overlaps with an SCO slot, it is not sent but the next inquiry message is awaited.

The messaging during the inquiry routines is summarized in Table 10.5 on page 112. In step 1, the master transmits an inquiry message using the inquiry access code and its own clock. The slave responds with the **FHS** packet which contains the slave's device address, native clock and other slave information. This **FHS** packet is returned at a semi-random time. The **FHS** packet is not acknowledged in the inquiry routine, but it is retransmitted at other times and frequencies as long as the master is probing with inquiry messages.

step	message	direction	hopping sequence	access code
1	ID	master to slave	inquiry	inquiry
2	FHS	slave to master	inquiry response	inquiry

Table 10.5: Messaging during inquiry routines.

If the scanning unit uses an optional scanning scheme, after responding to an inquiry with an FHS packet, it will perform page scan using the mandatory page scan scheme for $T_{mandatory\ pscan}$ period. Every time an inquiry response is sent the unit will start a timer with a timeout of $T_{mandatory\ pscan}$. The timer will be reset at each new inquiry response. Until the timer times out, when the unit performs page scan, it will use the mandatory page scanning scheme in the SR mode it uses for all its page scan intervals. Using the mandatory page scan scheme after the inquiry procedure enables all units to connect even if they do not support an optional paging scheme (yet). In addition to using the mandatory page scan scheme, an optional page scan scheme can be used in parallel for the $T_{mandatory\ pscan}$ period.

The $T_{mandatory\ pscan}$ period is included in the SP field of the FHS packet returned in the inquiry response routine, see Section 4.4.1.4 on page 56. The value of the period is indicated in the Table 10.6

SP mode	$T_{mandatory\ pscan}$
P0	≥20s
P1	≥40s
P2	≥60s
Reserved	-

Table 10.6: Mandatory scan periods for P0, P1, P2 scan period modes.

10.8 CONNECTION STATE

In the **CONNECTION** state, the connection has been established and packets can be sent back and forth. In both units, the channel (master) access code and the master Bluetooth clock are used. The hopping scheme uses the *channel hopping sequence*. The master starts its transmission in even slots (CLK_{1-0}=00), the slave starts its transmission in odd slots (CLK_{1-0}=10)

The **CONNECTION** state starts with a POLL packet sent by the master to verify the switch to the master's timing and channel frequency hopping. The slave can respond with any type of packet. If the slave does not receive the POLL packet or the master does not receive the response packet for *newconnectionTO* number of slots, both devices will return to **page/page scan** substates.

The first information packets in the **CONNECTION** state contain control messages that characterize the link and give more details regarding the Bluetooth units. These messages are exchanged between the link managers of the units. For example, it defines the SCO links and the sniff parameters. Then the transfer of user information can start by alternately transmitting and receiving packets.

The **CONNECTION** state is left through a **detach** or **reset** command. The **detach** command is used if the link has been disconnected in the normal way. All configuration data in the Bluetooth link controller is still valid. The **reset** command is a hard reset of all controller processes. After a reset, the controller has to be reconfigured.

The Bluetooth units can be in several modes of operation during the **CONNECTION** state: active mode, sniff mode, hold mode, and park mode. These modes are now described in more detail.

10.8.1 Active mode

In the active mode, the Bluetooth unit actively participates on the channel. The master schedules the transmission based on traffic demands to and from the different slaves. In addition, it supports regular transmissions to keep slaves synchronized to the channel. Active slaves listen in the master-to-slave slots for packets. If an active slave is not addressed, it may sleep until the next new master transmission. From the type indication in the packet, the number of slots the master has reserved for its transmission can be derived; during this time, the non-addressed slaves do not have to listen on the master-to-slave slots. A periodic master transmission is required to keep the slaves synchronized to the channel. Since the slaves only need the channel access code to synchronize with, any packet type can be used for this purpose.

10.8.2 Sniff mode

In the sniff mode, the duty cycle of the slave's listen activity can be reduced. If a slave participates on an ACL link, it has to listen in every ACL slot to the master traffic. With the sniff mode, the time slots where the master can start transmission to a specific slave is reduced; that is, the master can only start transmission in specified time slots. These so-called sniff slots are spaced regularly with an interval of T_{sniff}.

The slave has to listen at D_{sniff} slot every sniff period, T_{sniff} for a $N_{sniff\ attempt}$ number of times. If the slave receives a packet in one of the $N_{sniff\ attempt}$ RX slots, it should continue listening as long as it receives packets to its own AM_ADDR. Once it stops receiving packets, it should continue listening for $N_{sniff\ timeout}$ RX slots or remaining of the $N_{sniff\ attempt}$ number of RX slots, whichever is greater.

To enter the sniff mode, the master shall issue a sniff command via the LM protocol. This message will contain the sniff interval T_{sniff} and an offset D_{sniff}. The timing of the sniff mode is then determined similar as for the SCO links. In addition, an initialization flag indicates whether initialization procedure 1 or 2 is being used. The master uses initialization 1 when the MSB of the current master clock (CLK_{27}) is 0; it uses initialization 2 when the MSB of the current master clock (CLK_{27}) is 1. The slave shall apply the initialization method as indicated by the initialization flag irrespective of its clock bit value CLK_{27}. The master-to-slave sniff slots determined by the master and the slave shall be initialized on the slots for which the clock satisfies the following equation

$$CLK_{27\text{-}1} \bmod T_{sniff} = D_{sniff} \qquad \text{for initialization 1}$$

$$(\overline{CLK}_{27}, CLK_{26\text{-}1}) \bmod T_{sniff} = D_{sniff} \qquad \text{for initialization 2}$$

The slave-to-master sniff slot determined by the master and the slave shall be initialized on the slots after the master-to-slave sniff slot defined above. After initialization, the clock value CLK(k+1) for the next master-to-slave SNIFF slot is found by adding the fixed interval T_{sniff} to the clock value of the current master-to-slave sniff slot:

$$CLK(k+1) = CLK(k) + T_{sniff}$$

10.8.3 Hold mode

During the **CONNECTION** state, the ACL link to a slave can be put in a **hold** mode. This means that the slave temporarily does not support ACL packets on the channel any more (note: possible SCO links will still be supported). With the **hold** mode, capacity can be made free to do other things like scanning, paging, inquiring, or attending another piconet. The unit in **hold** mode can also enter a low-power sleep mode. During the **hold** mode, the slave unit keeps its active member address (AM_ADDR).

Prior to entering the hold mode, master and slave agree on the time duration the slave remains in the hold mode. A timer is initialized with the *holdTO* value. When the timer is expired, the slave will wake up, synchronize to the traffic on the channel and will wait for further master instructions.

10.8.4 Park mode

When a slave does not need to participate on the piconet channel, but still wants to remain synchronized to the channel, it can enter the park mode which is a low-power mode with very little activity in the slave. In the park mode, the slave gives up its active member address AM_ADDR. Instead, it receives two new addresses to be used in the park mode

- PM_ADDR: 8-bit Parked Member Address
- AR_ADDR: 8-bit Access Request Address

The PM_ADDR distinguishes a parked slave from the other parked slaves. This address is used in the master-initiated unpark procedure. In addition to the PM_ADDR, a parked slave can also be unparked by its 48-bit BD_ADDR. The all-zero PM_ADDR is a reserved address: if a parked unit has the all-zero PM_ADDR it can only be unparked by the BD_ADDR. In that case, the PM_ADDR has no meaning. The AR_ADDR is used by the slave in the slave-initiated unpark procedure. All messages sent to the parked slaves have to be carried by broadcast packets (the all-zero AM_ADDR) because of the missing AM_ADDR.

The parked slave wakes up at regular intervals to listen to the channel in order to re-synchronize and to check for broadcast messages. To support the synchronization and channel access of the parked slaves, the master supports a beacon channel described in the next section. The beacon structure is communicated to the slave when it is being parked. When the beacon structure changes, the parked slaves are updated through broadcast messages.

In addition for using it for low power consumption, the park mode is used to connect more than seven slaves to a single master. At any one time, only seven slaves can be active. However, by swapping active and parked slaves out respectively in the piconet, the number of slave virtually connected can be much larger (255 if the PM_ADDR is used, and even a larger number if the BD_ADDR is used). There is no limitation to the number of slaves that can be parked.

10.8.4.1 Beacon channel

To support parked slaves, the master establishes a beacon channel when one or more slaves are parked. The beacon channel consists of one beacon slot or a train of equidistant beacon slots which is transmitted periodically with a constant time interval. The beacon channel is illustrated in Figure 10.8 on page 117. A train of N_B ($N_B \geq 1$) beacon slots is defined with an interval of T_B slots.

The beacon slots in the train are separated by Δ_B. The start of the first beacon slot is referred to as the **beacon instant** and serves as the beacon timing reference. The beacon parameters N_B and T_B are chosen such that there are sufficient beacon slots for a parked slave to synchronize to during a certain time window in an error-prone environment.

When parked, the slave will receive the beacon parameters through an LMP command. In addition, the timing of the beacon instant is indicated through the offset D_B. Like for the SCO link (see Section 3.2 on page 45), two initialization procedures 1 or 2 are used. The master uses initialization 1 when the MSB of the current master clock (CLK_{27}) is 0; it uses initialization 2 when the MSB of the current master clock (CLK_{27}) is 1. The chosen initialization procedure is also carried by an initialization flag in the LMP command. The slave shall apply the initiations method as indicated by the initialization flag irrespective of its clock bit CLK_{27}. The master-to-slave slot positioned at the beacon instant shall be initialized on the slots for which the clock satisfies the following equation

$$CLK_{27\text{-}1} \bmod T_B = D_B \qquad \text{for initialization 1}$$

$$(\overline{CLK}_{27}, CLK_{26\text{-}1}) \bmod T_B = D_B \qquad \text{for initialization 2}$$

After initialization, the clock value CLK(k+1) for the next beacon instant is found by adding the fixed interval T_B to the clock value of the current beacon instant:

$$CLK(k+1) = CLK(k) + T_B$$

The beacon channel serves four purposes:

1. transmission of master-to-slave packets which the parked slaves can use for re-synchronization
2. carrying messages to the parked slaves to change the beacon parameters
3. carrying general broadcast messages to the parked slaves
4. unparking of one or more parked slaves

Since a slave can synchronize to any packet which is preceded by the proper channel access code, the packets carried on the beacon slots do not have to contain specific broadcast packets for parked slaves to be able to synchronize; any packet can be used. The only requirement placed on the beacon slots is that there is master-to-slave transmission present. If there is no information to be sent, **NULL** packets can be transmitted by the master. If there is indeed broadcast information to be sent to the parked slaves, the first packet of the broadcast message shall be repeated in every beacon slot of the beacon train. However, synchronous traffic like on the SCO link, may interrupt the beacon transmission.

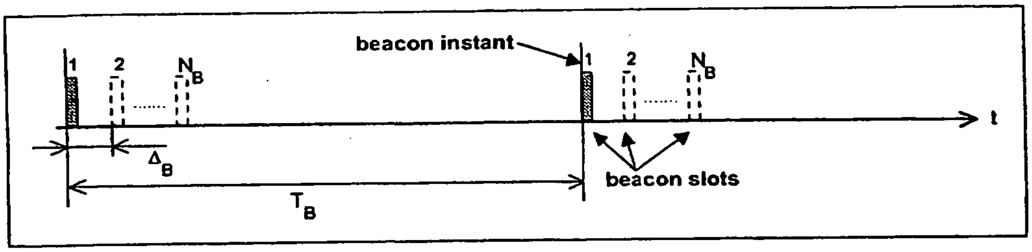

Figure 10.8: General beacon channel format

10.8.4.2 Beacon access window

In addition to the beacon slots, an access window is defined where the parked slaves can send requests to be unparked. To increase reliability, the access window can be repeated M_{access} times ($M_{access} \geq 1$), see Figure 10.9 on page 117. The access window starts a fixed delay D_{access} after the beacon instant. The width of the access window is T_{access}.

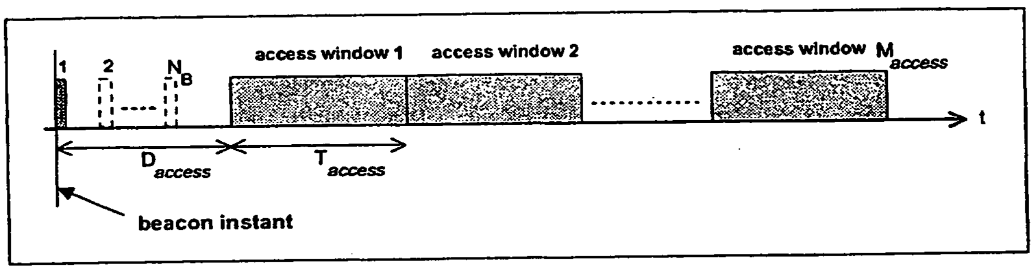

Figure 10.9: Definition of access window

The access window may support different slave access techniques, like polling, random access, or other forms of access. At this stage, only the polling technique has been defined. The format of the polling technique is shown in Figure 10.10 on page 118. The same TDD structure is used as on the piconet channel, i.e. master-to-slave transmission is alternated by slave-to-master transmission. The slave-to-master slot is divided into two half slots of 312.5 µs each. The half slot a parked slave is allowed to respond in corresponds to its access request address (AR_ADDR), see also section 10.8.4.6 on page 120. For counting the half slots to determine the access request slot, the start of the access window is used, see Figure 10.10 on page 118. The slave is only allowed to send an access request in the proper slave-to-master half slot if in the preceding master-to-slave slot a broadcast packet has been received. In this way, the master polls the parked slaves.

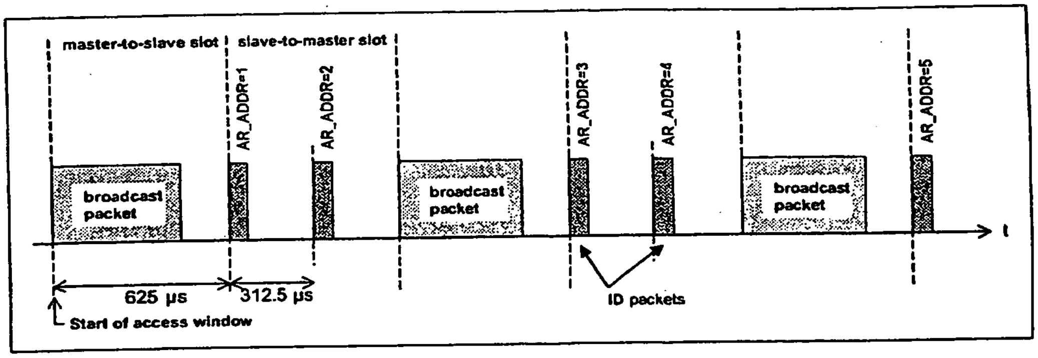

Figure 10.10: Access procedure applying the polling technique.

However, the slots of the access window can also be used for traffic on the piconet if required. For example, if an SCO connection has to be supported, the slots reserved for the SCO link may carry SCO information instead of being used for access requests, i.e. if the master-to-slave slot in the access window contains a packet different from a broadcast packet, the following slave-to-master slot cannot be used for slave access requests. Slots in the access window not affected by traffic can still be used according to the defined access structure; an example is shown in Figure 10.11 on page 118: the access procedure is continued as if no interruption had taken place.

When the slave is parked, it is indicated what type of access scheme will be used. For the polling scheme, the number of slave-to-master access slots N_{acc_slot} is indicated.

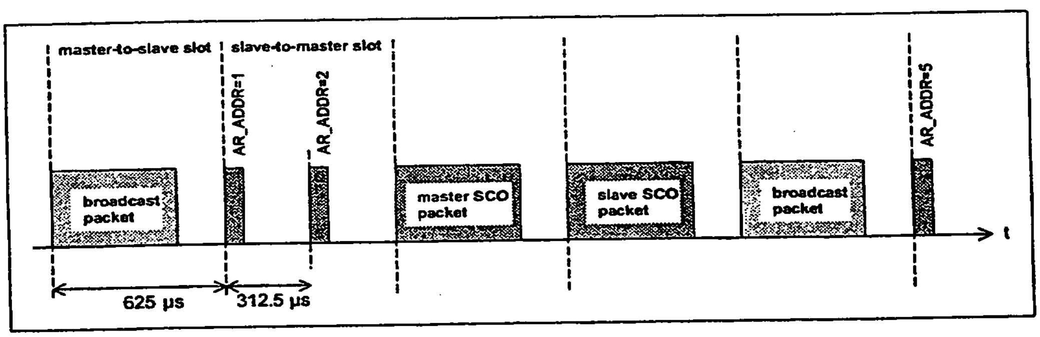

Figure 10.11: Disturbance of access window by SCO traffic

By default, the access window is always present. However, its activation depends on the master sending broadcast messages to the slave at the appropriate slots in the access window. A broadcast LMP command in the beacon slots may indicate that the access window following will not be activated. This prevents unnecessary scanning of parked slaves that want to request access.

10.8.4.3 Parked slave synchronization

Parked slaves sleep most of the time. However, periodically they wake up to re-synchronize to the channel. Any packet exchanged on the channel can be used for synchronization. Since master transmission is mandatory on the beacon slots, parked slaves will exploit the beacon channel to re-synchronize. A parked slave will wake-up at the beacon instant to read the packet sent on the first beacon slot. If this fails, it will retry on the next beacon slot in the beacon train; in total, there are N_B opportunities per beacon instant to re-synchronize. During the search, the slave may increase its search window, see also Section 9.4 on page 90. The separation between the beacon slots in the beacon train Δ_B is chosen such that consecutive search windows will not overlap.

The parked slave does not have to wake up at every beacon instant. Instead, a sleep interval can be applied which is longer than the beacon interval T_B, see Figure 10.12 on page 119. The slave sleep window must be a multiple N_{B_sleep} of T_B. The precise beacon instant the slave shall wake up on is indicated by the master with D_{B_sleep} which indicates the offset (in multiples of T_B) with respect to the beacon instant ($0< D_{B\_sleep}<N_{B\_sleep}-1$). To initialize the wake-up period, the following equations are used:

$$CLK_{27-1} \bmod (N_{B\_sleep} \bullet T_B) = D_B + D_{B\_sleep} \bullet T_B \qquad \text{for initialization 1}$$

$$(\overline{CLK_{27}}, CLK_{26-1}) \bmod (N_{B\_sleep} \bullet T_B) = D_B + D_{B\_sleep} \bullet T_B \quad \text{for initialization 2}$$

where initialization 1 is chosen by the master if the MSB in the current master clock is 0 and initialization 2 is chosen if the MSB in the current master clock is 1.

When the master wants to send broadcast messages to the parked slaves, it may use the beacon slots for these broadcast messages. However, if $N_B<N_{BC}$, the slots following the last beacon slot in the beacon train shall be used for the remaining $N_{BC}-N_B$ broadcast packets. If $N_B>N_{BC}$, the broadcast message is repeated on all N_B beacon slots.

A parked slave shall at least read the broadcast messages sent in the beacon slot(s) it wakes up in; the minimum wake-up activity is to read the channel access code for re-synchronization and the packet header to check for broadcast messages.

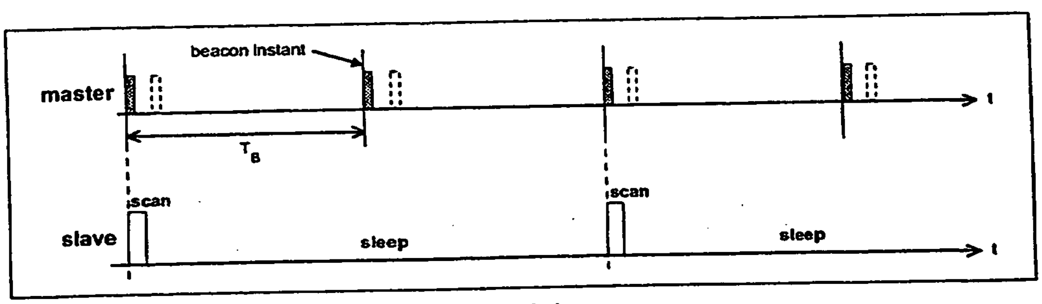

Figure 10.12: Extended sleep interval of parked slaves.

10.8.4.4 Parking

A master can park an active slave through the exchange of one or a few LMP commands. Before put into the park mode, the slave is assigned a PM_ADDR and an AR_ADDR. Every parked slave has a unique PM_ADDR; however, the AR_ADDR is not necessarily unique. Also, the beacon parameters are given by the master when the slave is parked. The slave then gives up its AM_ADDR and enters the park mode. A master can park only a single slave at a time. The park message is carried with a normal data packet and addresses the slave through its AM_ADDR.

10.8.4.5 Master-activated unparking

The master can unpark a parked slave by sending a dedicated LMP unpark command including the parked slave's address. This message is sent in a broadcast packet on the beacon slots. Either the slave's PM_ADDR is used, or its full BD_ADDR is used. The message also includes the active member address AM_ADDR the slave will use after it has re-entered the piconet. The unpark message can include a number of slave addresses so that multiple slaves can be unparked simultaneously. For each slave, a different AM_ADDR is assigned.

After having received the unpark message, the parked slave matching the PM_ADDR or BD_ADDR will leave the park mode and enter the active mode. It will keep listening to the master until it is addressed by the master through its AM_ADDR. The first packet sent by the master should be a POLL packet. The return packet in response to the POLL packet confirms that the slave has been unparked. If no response packets from the slave is received for *newconnectionTO* number of slots after the end of beacon repetition period, the master will unpark the slave again. If the slave does not receive the POLL packet for *newconnectionTO* number of slots after the end of beacon repetition period, it will return to park, with the same beacon parameters. After confirming that the slave is active, the master decides in which mode the slave will continue.

10.8.4.6 Slave-activated unparking

A slave can request access to the channel through the access window defined in section 10.8.4.2 on page 117. As shown in Figure 10.10 on page 118, the access window includes several slave-to-master half slots where the slave can send an access request message. The specific half slot the slave is allowed to respond in, corresponds to its access request address (AR_ADDR) which it has received when it was parked. The order of the half slots (in Figure 10.10 the AR_ADDR numbers linearly increase from 1 to 5) is not fixed: an LMP command sent in the beacon slots may reconfigure the access window. When a slave desires access to the channel, it sends an access request message in the proper slave-to-master half slot.The access request message of the slave is the **ID** packet containing the device access code (DAC) of the master (which is in this case the channel access code without the trailer). The parked slave is

only allowed to transmit an access request message in the half slot when in the preceding master-to-slave slot, a broadcast packet has been received. This broadcast message can contain any kind of broadcast information not necessarily related to the parked slave(s). If no broadcast information is available, a broadcast **NULL** or broadcast **POLL** packet shall be sent.

After having sent an access request, the parked slave will listen for an unpark message from the master. As long as no unpark message is received, the slave will repeat the access requests in the subsequent access windows. After the last access window (there are M_{access} windows in total, see Section 10.8.4.2 on page 117), the parked slave shall listen for an additional N_{poll} time slots for an unpark message. If no unpark message is received within N_{poll} slots after the end of the last access window, the slave may return to sleep and retry an access attempt after the next beacon instant.

After having received the unpark message, the parked slave matching the PM_ADDR or BD_ADDR will leave the park mode and enter the active mode. It will keep listening to the master until it is addressed by the master through its AM_ADDR. The first packet sent by the master should be a POLL packet. The return packet in response to the POLL packet confirms that the slave has been unparked. If no response packet from the slave is received for *newconnectionTO* number of slots after N_{poll} slots after the end of the last access window, the master will send the unpark message to the slave again. If the slave does not receive the POLL packet for *newconnectionTO* number of slots after N_{poll} slots after the end of the last access window, it will return to park, with the same beacon parameters. After confirming that the slave is active, the master decides in which mode the slave will continue.

10.8.4.7 Broadcast scan window

In the beacon train, the master can support broadcast messages to the parked slaves. However, it may extend its broadcast capacity by indicating to the parked slaves that more broadcast information is following after the beacon train. This is achieved by a special LMP command ordering the parked slaves (as well as the active slaves) to listen to the channel for broadcast messages during a limited time window. This time window starts at the beacon instant and continues for the period as indicated in the LMP command sent in the beacon train.

10.8.5 Polling schemes

10.8.5.1 Polling in active mode

The master always has full control over the piconet. Due to the stringent TDD scheme, slaves can only communicate with the master and not to other slaves. In order to avoid collisions on the ACL link, a slave is only allowed to transmit in the slave-to-master slot when addressed by the AM_ADDR in the packet

header in the preceding master-to-slave slot. If the AM_ADDR in the preceding slot does not match, or an AM_ADDR cannot be derived from the preceding slot, the slave is not allowed to transmit.

On the SCO links, the polling rule is slightly modified. The slave is allowed to transmit in the slot reserved for his SCO link unless the (valid) AM_ADDR in the preceding slot indicates a different slave. If no valid AM_ADDR can be derived in the preceding slot, the slave is still allowed to transmit in the reserved SCO slot.

10.8.5.2 Polling in park mode

In the park mode, parked slaves are allowed to send access requests in the access window provided a broadcast packet is received in the preceding master-to-slave slot. Slaves in active mode will not send in the slave-to-master slots following the broadcast packet since they are only allowed to send if addressed specifically.

10.8.6 Slot reservation scheme

The SCO link is established by negotiations between the link managers which involves the exchange of important SCO timing parameters like T_{SCO} and D_{SCO} through LMP messages.

10.8.7 Broadcast scheme

The master of the piconet can broadcast messages which will reach all slaves. A broadcast packet is characterized by the all-zero AM_ADDR. Each new broadcast message (which may be carried by a number of packets) shall start with the flush indication (L_CH=10).

A broadcast packet is never acknowledged. In an error-prone environment, the master may carry out a number of retransmissions N_{BC} to increase the probability for error-free delivery, see also Section 5.3.5 on page 72.

In order to support the **park** mode (as described in Section 10.8.4 on page 115), a master transmission shall take place at fixed intervals. This master transmission will act as a beacon to which slaves can synchronize. If no traffic takes place at the beacon event, broadcast packets shall be sent. More information is given in Section 10.8.4 on page 115.

10.9 SCATTERNET

10.9.1 General

Multiple piconets may cover the same area. Since each piconet has a different master, the piconets hop independently, each with their own channel hopping

sequence and phase as determined by the respective master. In addition, the packets carried on the channels are preceded by different channel access codes as determined by the master device addresses. As more piconets are added, the probability of collisions increases; a graceful degradation of performance results as is common in frequency-hopping spread spectrum systems.

If multiple piconets cover the same area, a unit can participate in two or more overlaying piconets by applying time multiplexing. To participate on the proper channel, it should use the associated master device address and proper clock offset to obtain the correct phase. A Bluetooth unit can act as a slave in several piconets, but only as a master in a single piconet: since two piconets with the same master are synchronized and use the same hopping sequence, they are one and the same piconet. A group of piconets in which connections consists between different piconets is called a **scatternet**.

A master or slave can become a slave in another piconet by being paged by the master of this other piconet. On the other hand, a unit participating in one piconet can page the master or slave of another piconet. Since the paging unit always starts out as master, a master-slave role exchange is required if a slave role is desired. This is described in the section 10.9.3 on page 123.

10.9.2 Inter-piconet communications

Time multiplexing must be used to switch between piconets. In case of ACL links only, a unit can request to enter the **hold** or **park** mode in the current piconet during which time it may join another piconet by just changing the channel parameters. Units in the **sniff** mode may have sufficient time to visit another piconet in between the sniff slots. If SCO links are established, other piconets can only be visited in the non-reserved slots in between. This is only possible if there is a single SCO link using **HV3** packets. In the four slots in between, one other piconet can be visited. Since the multiple piconets are not synchronized, guard time must be left to account for misalignment. This means that only 2 slots can effectively be used to visit another piconet in between the **HV3** packets.

Since the clocks of two masters of different piconets are not synchronized, a slave unit participating in two piconets has to take care of two offsets that, added to its own native clock, create one or the other master clock. Since the two master clocks drift independently, regular updates of the offsets are required in order for the slave unit to keep synchronization to both masters.

10.9.3 Master-slave switch

In principle, the unit that creates the piconet is the master. However, a master-slave (MS) switch can take place when a slave wants to become a master. For the two units involved in the switch, the MS switch results in a reversal of their TX and RX timing: a <u>TDD switch</u>. However, since the piconet parameters are derived from the device address and clock of the master, a master-slave switch inherently involves a redefinition of the piconet as well: a <u>piconet switch</u>. The

new piconet's parameters are derived from the former slave's device address and clock. As a consequence of this piconet switch, other slaves in the piconet not involved in the switch have to be moved to the new piconet, changing their timing and their hopping scheme. The new piconet parameters have to be communicated to each slave. The scenario to achieve this is described below. Assume unit A wants to become master; unit B was the former master. The following steps are taken.

- Slave A and master B agree to exchange roles.
- When confirmed by both units, both slave A and master B do the TDD switch but keep the former hopping scheme (still using the device address and clock of unit B), so there is no piconet switch yet.
- Unit A is now the master of the piconet. Since the old and new masters' clocks are asynchronous, the 1.25 ms resolution of the clock information given in the FHS packet is not sufficient for aligning the slot boundaries of the two piconets. Prior to sending the FHS packet, the new master A sends an LMP packet giving the delay between the start of the master-to-slave slots of the old and new piconet channels. This timing information ranges from 0 to 1249 μs with a resolution of 1 μs. It is used together with the clock information in the FHS packet to accurately position the correlation window when switching to the new master's timing after acknowledgment of the FHS packet.
- After the time alignment LMP message, Master A sends an FHS packet including the new AM_ADDR to slave B (the AM_ADDR in the FHS packet header is the all-zero address) still using the "old" piconet parameters. After the FHS acknowledgement, which consists of the **ID** packet and is sent by the slave on the old hopping sequence, both master A and slave B turn to the new channel parameters of the new piconet as indicated by the FHS and time alignment LMP packets (at least for the A-B connection).
- A piconet switch is enforced on each slave separately. Master A sends a time alignment and an FHS packets and waits for an acknowledgement. Transmission of the FHS packet and the acknowledgement continues on the "old" piconet parameters of unit B (compare this to the page hopping scheme used during connection establishment, see Section 10.6.4 on page 104). After FHS acknowledgement using an **ID** packet sent by the slave, the communication to this slave continues with the new device address and clock of unit A. The FHS packet sent to each slave has the old AM_ADDR in the FHS packet header and their new AM_ADDR in the FHS packet payload (the new AM_ADDR may be identical to the old AM_ADDR).
- After reception of the FHS packet acknowledgement, the new master A switches to its own timing and sends a POLL packet to verify the switch. Both the master and the slave will start a timer with a time out of *newconnectionTO* on FHS packet acknowledgement. If no response is received, the master resends the POLL packet until *newconnectionTO* is reached. After this timeout both the slave and the master return to the old piconet timing (but the TDD switch remains).The master sends the FHS packet again and the procedure is repeated.

- The new master repeats the above procedure for each slave in the old piconet.

Summarized, the MS-switch takes place in two steps: first a TDD switch of the considered master and slave, and then a piconet switch of all participants. When all slaves have acknowledged the reception of the FHS packet, each unit uses the new piconet parameters defined by the new master and the piconet switch is a fact. The information on the AM_ADDR, PM_ADDR, and other features of the old slaves is transferred from the old master to the new master. The transfer procedure is outside the scope of this procedure. Parked slaves shall be activated (using the old park parameters), be changed to the new piconet parameters, and then return to the **park** mode using the new park parameters.

10.10 POWER MANAGEMENT

Features are included into Bluetooth to ensure a low-power operation. These features are both at the microscopic level when handling the packets, and at the macroscopic level using certain operation modes.

10.10.1 Packet handling

In order to minimize power consumption, packet handling is minimized both at TX and RX sides. At the TX side, power is minimized by only sending useful data. This means that if only link control information needs to be exchanged, **NULL** packets will be used. No transmission is carried out at all if there is no link control information or involves a NAK only (NAK is implicit on no reply). If there is data to be sent, the payload length is adapted in order to send only the valid data bytes. At the RX side, packet processing takes place in different steps. If no valid access code is found in the search window, the transceiver returns to sleep. If an access code is found, the receiver unit is woken up and starts to process the packet header. If the HEC fails, the unit will return to sleep after the packet header. A valid header will indicate if a payload will follow and how many slots are involved.

10.10.2 Slot occupancy

As was described in Section 4.4 on page 54, the packet type indicates how many slots a packet may occupy. A slave not addressed in the first slot can go to sleep for the remaining slots the packet may occupy. This can be read from the TYPE code.

10.10.3 Low-power modes

In Section 10.8 on page 112, three modes were described during the **CONNECTION** state which reduce power consumption. If we list the modes in increasing order of power efficiency then the **sniff** mode has the higher duty

cycle, followed by the **hold** mode with a lower duty cycle, and finishing with the **park** mode with the lowest duty cycle.

10.11 LINK SUPERVISION

A connection may break down due to various reasons such as a device moving out of range or a power failure condition. Since this may happen without any prior warning, it is important to monitor the link on both the master and the slave side to avoid possible collisions when the AM_ADDR is reassigned to another slave.

To be able to supervise link loss, both the master and the slave use link supervision timers, $T_{supervision}$. Upon reception of a packet that passes the HEC check and has the correct AM_ADDR, the timer is reset. If at any time in connection state, the timer reaches the *supervisionTO* value, the connection is reset. The same timeout value is used for both SCO and ACL connections.

The timeout period, *supervisionTO*, is negotiated at the LM level. Its value is chosen so that the supervision timeout will be longer than hold and sniff periods. Link supervision of a parked slave will be done by unparking and re-parking the slave.